

Logic-System Design Techniques Reduce Switching-CMOS Power

By adopting certain techniques in the design of your CMOS-based logic system, you can effect dramatic reductions in the transitional power these zero-quiescent-current devices consume when switching.

This article describes ways to reduce the power consumption in logic designs using high-speed CMOS ICs. The MM74HC logic family has near-zero power dissipation when in the quiescent mode. Its only substantial power drain arises from dynamic switching currents. Traditional TTL and NMOS systems do not share this low-power feature, requiring instead that you reduce power by selecting low-power ICs and external components.

The CMOS device is inherently efficient, but you can greatly enhance system efficiency by designing around the following guidelines:

- minimizing effective system operating frequency;
- minimizing static DC current paths (e.g., in pull-up or pull-down resistors);
- putting the logic to sleep (by removing the clock);
- capitalizing on power-down situations.

Total system power dissipation is the sum of two components: static (or quiescent) and dynamic power. LS TTL systems consume such a great amount of quiescent power that the dynamic component pales into insignificance. When using MM74HC logic in power-critical applications, however, you must consider both components. The following sections describe how to determine system power by using MM74HC devices' power-dissipation-capacitance (C_{PD}) specifications. The text also discusses a few power-reduction philosophies and some of the differences in consumption for MM74HCT TTL-compatible CMOS logic. Because system power is simply total I_{CC} times the supply voltage, the calculations treat power and current interchangeably.

Calculating the quiescent power is just as easy—the sum of the DC currents times the supply voltage. Thus, total system quiescent power is

$$P_{\text{SYSTEM}} = (I_{CC1} + I_{CC2} + \dots + I_{CCn}) V_{CC} \quad (1)$$

The currents in this expression are caused by pull-up and load resistors and TTL, NMOS and linear circuits in the system. If it's appreciable—although unlikely—you can include the very small quiescent I_{CC} of MM74HC devices. Generally, the worst-case I_{CC} values in the CMOS ICs' datasheets are very conservative. Typical values range from ten to 100 times less than the limits; moreover, it's almost statistically impossible for a system to contain all worst-case devices.

As pointed out earlier, the major contributors to CMOS ICs' power dissipation are dynamic switching currents. Figure 1 is a schematic diagram of one MM74HC00 NAND gate, and it shows the dynamic currents that result from switching one input LOW-to-HIGH. When the IC is not switching, there's no DC current path from V_{CC} to ground except for leakage. This is because whenever an n-channel device is ON, its complementary p-channel partner is OFF.

CMOS power consumption is caused by the transient currents that charge and discharge internal and external capacitances during logic transitions. As frequency increases, these currents naturally increase. You can't measure these currents or their associated capacitances individually, but you can measure the total current. You can equate this total current to a power-dissipation capacitance (C_{PD}) as follows:

$$I_{CC} = (C_{PD} + C_L)(V_{CC})(f_{IC}) \quad (2)$$

where I_{CC} is the supply current, V_{CC} is the supply voltage, f_{IC} is the input toggle rate and C_L is the toggled load capacitance. Referring again to Figure 1, the load current I_L results from switching the load capacitance. To obtain the internal equivalent capacitance, you must subtract the load current from I_{CC} .

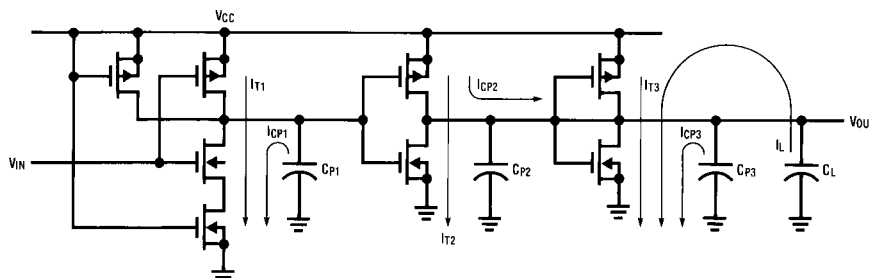


FIGURE 1. Principal contributors to CMOS power consumption, these transient currents are the result of transitional charging and discharging of internal and load capacitances. The average currents are naturally a function of the operating frequency.

Using the C_{PD} figure specified in datasheets, you can estimate the current consumption of each device in your system if you know the toggling frequency. By multiplying both sides of Equation 2 by V_{CC} , you can determine the dynamic power consumption.

$$P_{DYNAMIC} = (C_{PD} + C_L)(V_{CC}^2)(f). \quad (3)$$

As mentioned, C_{PD} is an indirect measure of the amount of switching current a circuit consumes. It depends on how much of the circuit's internal logic is switching and how many outputs are toggling. For example, a MM74HC374 octal 3-STATE flip-flop clocked at 1 MHz dissipates much more power if its data inputs change every clock period than it would if its outputs are disabled and its inputs are tied HIGH or LOW during clocking.

Figure 3 shows that when the flip-flop's outputs are enabled and the data inputs are changing, virtually all internal nodes are toggling and all internal parasitic capacitances are charging. On the other hand, if the data is held HIGH and the outputs are disabled, only the clock logic dissipates power (and very little at that). All other sections are static.

As you'll see, the method of testing C_{PD} (see "Test C_{PD} in realistic situations") can yield various values that might or might not be applicable to the particular way the part is being used. Fortunately, several generalizations allow reasonable approximations to C_{PD} 's value, as discussed in the following section.

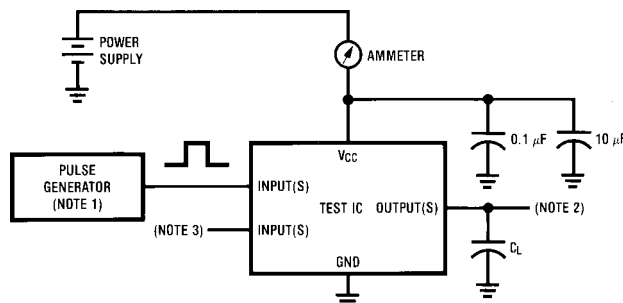
Test C_{PD} in Realistic Situations

In MM74HC datasheets, one or two C_{PD} values are specified. At best, the parameter is a simplification of the worst-case operating mode of a device under typical operating conditions. However, because most devices have several possible toggling modes (each having a different power consumption), you might do well to characterize C_{PD} for your particular application.

The nearby Figure 2 shows a circuit for measuring C_{PD} . Normally, the IC is set up in a given toggling mode, with its output pins pulled out of the test socket to reduce stray-induced errors. For automated testing, you could use a standard load (e.g., 50 pF) and subtract its I_{CC} contribution from the total. The ammeter in series with the V_{CC} line is bypassed with 0.1- and 1- μ F capacitors.

For simple measurements, you can set the input's toggle frequency at 200 kHz, with $V_{CC} = 5V$. This yields an ammeter reading in microamps that's equal to C_{PD} in picofarads. You could use other voltages and frequencies, but little variation should result. For example, JEDEC's high-speed-CMOS committee recommends 1 MHz.

To better understand what datasheet C_{PD} means, the following listing describes by part type how each IC is toggled. In measuring C_{PD} , the worst path is always chosen. Moreover, within the constraints listed, as much of the internal circuitry and as many of the outputs as possible are toggled simultaneously.



Notes:

1. OUTPUT = square wave with ≤ 6 -nsec rise and fall times; levels = GND and V_{CC} .
2. Bend all output pins from test socket, or use known load and deduct its current from measured I_{CC} .
3. Terminate all unused inputs to GND or V_{CC} .

FIGURE 2. Measure equivalent CMOS-system capacitance with this simple test circuit.
The text describes how to toggle the various CMOS logic functions (excepting one-shots, of course, which draw DC power).

- **Gates:** All inputs except one are held at either V_{CC} or ground, depending on which state causes the output to toggle. The one remaining input is toggled at a given frequency. C_{PD} is given on a per-gate basis.
- **Decoders:** One input is toggled, thereby causing the outputs to toggle at the same rate. Normally, one of the address-select pins is switched while the decoder is enabled. All other inputs are tied to V_{CC} or ground, whichever enables operation. C_{PD} is expressed on a per-independent-decoder basis.
- **Multiplexers:** One data input is tied HIGH, and a second is tied LOW. The address-select lines and enable inputs are configured such that by toggling one address line the two data inputs are alternately selected, causing the outputs to toggle. If it's a 3-STATE Multiplexer, C_{PD} is given for outputs both enabled and disabled. C_{PD} is measured per multiplexer function.
- **3-STATE buffers and transceivers:** When the outputs are enabled, C_{PD} is measured as for simple gates; i.e., on a per-buffer basis. The same holds true for the 3-STATE condition. Transceivers are measured per buffer as well, both enabled and disabled.

Test CPD in Realistic Situations (Continued)

- **Latches:** The device is clocked and data is toggled every other clock pulse. Other preset or clear inputs are held to enable output toggling. If the device has commonly clocked latches, the clock is toggled and one latch is exercised. 3-STATE latches are measured with their outputs both enabled and disabled. C_{PD} is given on a per-latch basis.
- **Flip-flops:** The same as for latches. The device's inputs are configured to toggle, and any preset or clear inputs are held inactive.
- **Shift registers:** The register is clocked and the serial data input is toggled every other clock pulse, as for latches and flip-flops. Other clear or load pins are held inactive, and parallel data inputs are held at V_{CC} or ground. 3-STATE devices are measured with outputs both enabled and disabled. If the device takes parallel loads only, it's loaded with 10101010... and clocked to shift the data out, then reloaded.
- **Counters:** A signal is applied to its clock input; other clear or load inputs are held inactive. C_{PD} is given for each counter within a package.
- **Arithmetic circuits:** Adders, magnitude comparators, encoders, parity generators, ALUs and other miscellaneous circuits. The general rule is to exercise these parts to obtain the maximum number of outputs toggling simultaneously while toggling only one or two inputs.
- **Display drivers:** C_{PD} is generally not required for LED drivers, because the LEDs use so much more power they overshadow the drivers' C_{PD} ; moreover, when blanked the drivers are rarely driven at any significant speed. If needed, however, C_{PD} is measured with outputs enabled and disabled, while toggling between a lamp test and blank (if provided), or between a display of numbers 6 and 7. LCD drivers are tested by toggling their phase inputs, which control the segment and backplane waveforms. If either of these driver types has latched inputs, the latches are set to a flow-through mode.
- **One-shots:** In some cases, when a device's I_{CC} is significant, C_{PD} might not be specified. When it is, C_{PD} is tested by toggling one trigger input such that the output is a square wave. The timing resistor is tied to a separate V_{CC} line, to eliminate its power contribution.

Figuring Dynamic System Power

How do you calculate a system's dynamic power? You can do it on several levels, depending on the accuracy needed. The simplest approach is to use a C_{PD} model that's the sum of the CMOS ICs' C_{PD} s and the load capacitances. Then, assuming an average frequency, plug these numbers into Equation 2 or Equation 3.

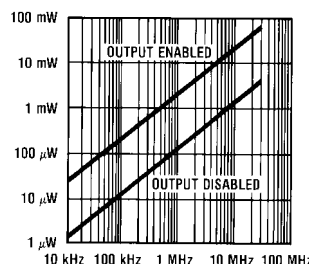


FIGURE 3. Output status determines dynamic dissipation in this 3-STATE-output flip-flop. The IC dissipates an order-of-magnitude higher power with its outputs enabled.

The most accurate approach, however, is to determine each component's operating frequency and its capacitive load. This method is used in critical battery powered applications. The following section describes this approach and proposes simplifications. In this approach, system dynamic power is the sum of the individual circuits' power dissipation:

$$P_T = P_1 + P_2 + P_3 + P_4 + \dots \text{etc.}, \quad (4)$$

where P_T is the total power and P_n is the power for each component. By substituting Equation 3 into Equation 4, the total system power is

$$P_T = (C_{P1} + C_{L1})(V_{CC}^2)(f) + (C_{P2} + C_{L2})(V_{CC}^2)(f) + \dots \text{etc.} \quad (5)$$

In Equation 5, load capacitances C_{L1} , C_{L2} , etc. are not simply the sum of all individual output loads. C_L is actually dependent on device type. Why? Different devices switch a different number of outputs simultaneously. What's more, these outputs can toggle at a different rate from that of the IC's clock or input. Thus, for an individual IC and its load, the actual power is

$$P_{IC} = V_{CC}^2 [(C_{PD}f) + (C_{L1}f_{L1}) + (C_{L2}f_{L2}) + \dots], \quad (6)$$

where C_L is the load on each of the simultaneously toggling outputs, and f_L is the toggle rate seen by the load. A good example is the power dissipation of a 4-bit CMOS counter. Here there are four output terms—each output switches at a different frequency. Accordingly, there are four (each) distinct C_L and f_L terms. To simplify Equation 6, define an effective load capacitance C_{LE} which is the actual load multiplied by the ratio of the load toggle rate to the IC's toggle frequency:

$$C_{LE} = (C_L)(f_L/f). \quad (7)$$

Substituting Equation 7 into Equation 6 and grouping terms,

$$P_{IC} = V_{CC}^2 f(C_{PD} + C_{LE1} + C_{LE2} + \dots). \quad (8)$$

This procedure simplifies the process because output toggle rates are almost exclusively a binary division of the input clock. Thus, for an accurate calculation of system power, you must calculate it for each IC using Equation 8 and take the total. The counter is a prime candidate for using Equation 8. Here, the first stage's effective output capacitance is half the actual; the second, one-quarter, and so on.

Tailor f, C to Device Type

To make practical use of the foregoing methods, the following list describes most of the CMOS-logic categories in terms of effective load and operating frequency:

- **Gates and buffers:** Power calculations for these are straightforward. C_{PD} , given for each gate, sums directly with its output load. Operating frequency is the rate at which the output toggles. For disabled 3-STATE buffers, the power calculation uses the 3-STATE-output C_{PD} multiplied by the input frequency (no load capacitance included.)
- **Decoders:** Each independent decoder can toggle no more than two outputs at a time. To calculate power consumption, sum C_{PD} with the load on two outputs. The frequency is the rate at which the outputs switch.
- **Multiplexers:** For non-3-STATE devices, sum the loads on all used outputs and add the sum to C_{PD} . The frequency is that at which the outputs switch. For 3-STATE devices, use only C_{PD} ; the frequency is the inputs' toggle rate.
- **Counters:** The operating frequency for each of a counter's outputs is that of the previous stage divided by two. The loads on lower order stages contribute less current. So to calculate power, sum C_{PD} with one-half the first stage's load plus one-quarter the second stage's, and so on. For decade and other modulo counters, this procedure is slightly different. In general, you can neglect outputs more than four stages removed from the clock. A simple approximation is to sum C_{PD} with the average output load and use the input clock frequency.
- **Latches, flip-flops and shift registers:** For these devices, the frequency is the ICs' clock rate. The outputs typically change state at half the clock rate, so when calculating power dissipation, add C_{PD} to half the output load. If the data inputs change more slowly, you can modify the effective load downward by the ratio of the data rate to the clock rate. Again, if the outputs are disabled, no load dissipation exists and you should use the 3-STATE C_{PD} .

These rules notwithstanding, it's rarely necessary to go through a detailed analysis of each IC. In most instances, a simpler analysis can yield good results. In noncritical applications where power consumption is used to determine the system's power-supply needs, the simpler analysis suffices. Using this method, you estimate the average operating frequency for major sections of the system. Next, sum all the C_{PD} s and effective loads in each section:

$$P_{BLOCK} = V_{CC}^2 f_{AVG} [(C_{P1} + C_{LE1}) + (C_{P2} + C_{LE2}) + \dots + (C_{Pn} + C_{LEn})]. \quad (9)$$

Thus, to approximate the total system's power consumption, you must approximate the effective loads for each group of devices (or the entire system) and add them together.

Consider a microprocessor-based system using an 8 MHz clock frequency. In this example, you might determine that the bus operates at approximately 2 MHz, random control logic at 4 MHz, and the RAM and I/O devices at 100 kHz. You could estimate an overall system clock to be 1 to 2 MHz, depending on the actual size of each block. Next, you'd sum the C_{PD} and the effective load capacitances—say 2000 and 1000 pF, respectively. The ballpark estimate for system power is

$$P = (5)^2 (1 \text{ MHz})(2000 \text{ pF} + 1000 \text{ pF}) = 75 \text{ mW}. \quad (10)$$

Exceptions to the above rules are one-shot ICs and gates configured as oscillators, which use CMOS in an essentially linear manner. Their power consumption is not strictly attributable to negligible quiescent currents or dynamic switching currents.

Consider one-shots, some of which draw DC current continuously, some only when the output pulse is triggered (check datasheets for the device type you're using). The culprits are the ICs' internal linear CMOS comparators that use DC bias circuits. HC one-shots use several design approaches. One (the MM74HC123A/221A/423A) uses a comparator that shuts off after a pulse times out; the second (the MM74HC4538) leaves the comparators on at all times.

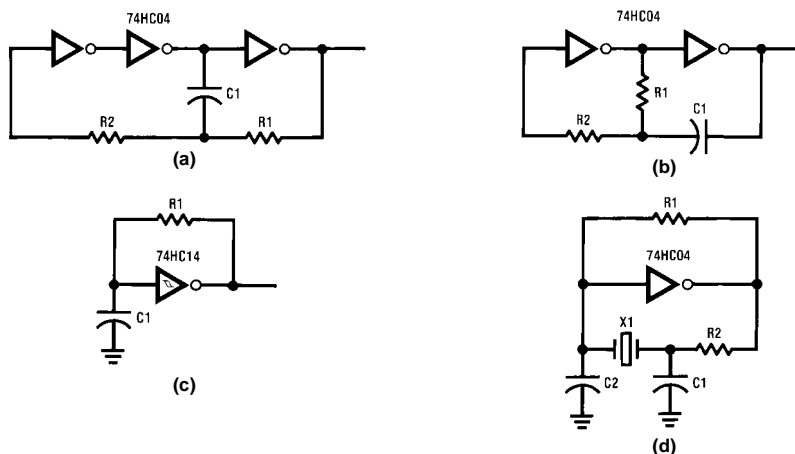


FIGURE 4. Drawing higher-than-calculated power, these CMOS oscillator configurations suffer from "soft" logic levels at their gates' inputs. Circuits (a) through (d) are 3-inverter, 2-inverter, Schmitt-trigger and crystal oscillators, respectively.

Tailor f, C to Device Type (Continued)

A one-shot's overall power consumption is its quiescent power plus the power consumed by its timing elements and C_{PD} . If the comparators turn off, you multiply the quiescent current by the duty cycle of the output pulse. Thus, the overall expression for one-shot power consumption is

$$P_{OS} = (I_{CC})(V_{CC})(D) + (C_{EXT} + C_L + C_{PD})(V_{CC}^2)(f), \quad (11)$$

where P_{OS} is the total power, D the one-shot's duty cycle, C_{EXT} the timing capacitor, C_L the load on both outputs, and f the operating frequency. In general, the C_{PD} term is small at lower frequencies; you can safely set it to zero to simplify the equation.

What about oscillators? The circuits shown in Figure 4 draw more current at a given operating frequency than you'd calculate using only C_{PD} . This is because in these applications, the inputs to some of the gates are at "soft" logic levels for significant amounts of time. This causes both p- and n-channel transistors to conduct simultaneously and hence draw DC current.

Figure 5(a) plots current vs. input voltage for the MM74HC00 gate and gives an idea of the amount of current typically drawn when soft logic levels are applied. The large spike at 2.3V is the result of the output's switching. At low frequencies, the oscillator's supply current can be several milliamps higher than you might expect because of the amount of DC current drawn.

The same is true of a MM74HC14 used as an oscillator. Figure 5(b) shows the supply current vs. input voltage for the MM74HC14 and the MM74C14 (or CD40106). Because the actual power consumed varies with frequency and component values, it's best to determine it empirically. As with the one-shots, the oscillator timing capacitor's contribution to power dissipation can be expressed by $P = V_{CC}(C_T)f$.

MM74HC logic uses bigger devices and lower transistor thresholds than metal-gate CMOS, so it might be more desirable to use either CD4000 or MM74C logic for lower power oscillators (if operating frequency and output-drive requirements permit.)

More Special Cases: HCT

Because of their unique applications in TTL and NMOS systems, MM74HCT devices have some additional traits that you should consider in designing systems. In TTL systems, the HCT ICs' inputs are driven under worst-case conditions by TTL levels of 0.5 and 2.4V. With these input levels applied, HCT consumes significant quiescent current: about 200 to 500 μA per input. You must consider this DC current when calculating power.

To see the origins of this quiescent current, refer to Figure 6, which shows a typical HCT's input. With a 2.4V input level, the n-channel transistor turns fully on; the p-channel device turns slightly on. This scenario results in a quiescent current dependent on the number of logic-One inputs applied. The 0.5V level is close enough to ground to cause the n-channel transistor to turn off, so HCT ICs draw quiescent current only when its inputs are at a high state.

The I_{CC} values with these logic levels are specified in the HCT datasheets. It's specified on a per-input basis—this allows you some flexibility in determining quiescent power when an IC is driven by both CMOS and TTL. The specified quiescent-current value results in calculated I_{CC} values

of several milliamps per IC, significantly less than that of LS TTL circuits.

Note, however, that using this datasheet approach yields current values roughly five times higher than that actually seen in system designs. The reason for this is that the I_{CC} test is specified at $V_{CC} = 5.5V$ and $V_{IN} = 2.4V$, but even worst-case TTL output-HIGH levels are at least 3.4V under these conditions. Output levels can only attain a low 2.4V with $V_{CC} = 4.5V$. Moreover, both TTL and NMOS outputs typically assume levels closer to 3V (at $V_{CC} = 4.5V$), lowering quiescent current more. The point is, don't let I_{CC} specs scare you into thinking CMOS is a power hog.

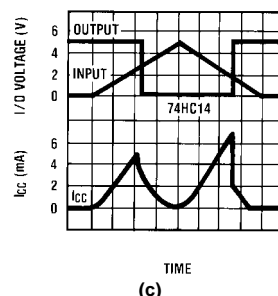
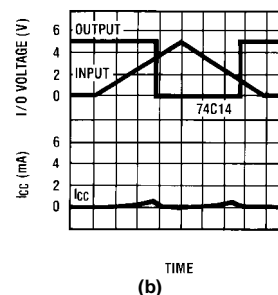
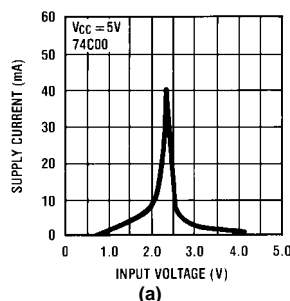


FIGURE 5. "Soft" logic levels cause high currents in a MM74HC00 inverter (a) and a MM74HC14 connected as an oscillator (b, c). Because the power varies with frequency and component values, it's best to determine its value empirically.

In mixed TTL-CMOS applications, the calculation of power consumed by the HCT logic must take into account both the dynamic and the quiescent currents. The dynamic portion is the same as that for HC logic—in fact, C_{PD} is measured with 0V and 5V input levels to exclude any quiescent

More Special Cases: HCT (Continued)

current. The static portion is the sum of the number of TTL logic-One inputs times their High-period duty cycle times the current per input. For a single IC, the power consumption is

$$P_{IC} = (V_{CC})(I_{CC})(N)(D) + V_{CC}^2 f(C_{PD} + C_{LE}), \quad (12)$$

where C_{LE} is defined as before, N is the number of TTL-driven inputs and D is the logic-High duty cycle.

I_{CC} is the datasheet's per-input spec. This expression can then be one term in Equation 4. If you're using the package-level quiescent current, the terms N and D drop out.

What about a situation in which HC drives HCT? In this scenario, ground and V_{CC} levels are applied, thereby ensuring that the p- and n-channel transistors don't turn on simultaneously. You can thus determine HCT power dissipation just as for HC by using C_{PD} .

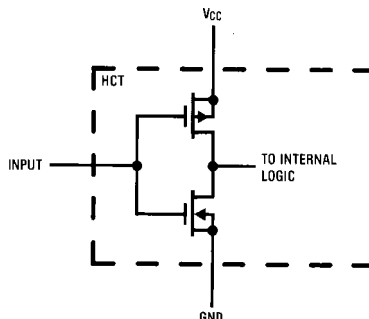


FIGURE 6. TTL-compatible CMOS is a special case. This schematic shows the MM74HCT family's input buffer. With a 2.4V input applied, the n-channel transistor is fully On; the p-channel, slightly On.

Now Let's Reduce Power

When designing low-power CMOS systems, there are several ways to minimize power. These methods involve reducing operating frequencies, cutting system load capacitances, using fast input transition times and minimizing any DC current paths.

First, for low-power system implementations, it's important not to over-design the operating frequency. Very simply put, it makes no sense to clock a counter at 20 MHz when 5 MHz will suffice.

Note that a reduction in overall system clock frequency doesn't necessarily entail a reduction in throughput. For example, consider a system consisting of four subsections, clocked at 8 MHz Figure 7(a). Rather than clocking all sections in parallel, you can reduce power by clocking each section only as fast as need be Figure 7(b). A second example of reducing the overall system clock rate is shown in Figure 8.

In (a), a CMOS memory array is driven directly from the CPU's address bus. Here, every memory is driven at the bus frequency. If, however, the address is latched by each memory block only when that block is being accessed (b), then only the block currently being accessed is clocked. This is why some CMOS RAMs incorporate on-chip address latches.

Another way to operate a system at the minimum possible frequency is to switch the system clock. The system is thus made to operate at the highest frequency only when

needed. Figure 9 shows the logic used to implement this scheme for a CMOS μP system. In this method, there are two oscillators, either of which can feed a divide-by-two circuit that provides a square-wave output. The flip-flop's output is the system clock. The system's μP can set or reset the flip-flop so that it can operate at either frequency.

Besides frequency reduction, there are several other methods to save power, including reducing load capacitances. You can accomplish this by reducing wiring capacitance (especially in high-frequency sections) through good layout practices, and by maintaining close proximity between interrelated high-frequency sections. In some instances where you might instinctively parallel several unused inputs, you can achieve lower load capacitance by tying the unused inputs to a supply. In another example, when using RC oscillators, it's best to use the smallest capacitor and the largest resistor possible.

Slow input transitions can cause extra dissipation. If an input signal rises slowly, it causes both input transistors to conduct for a longer time, thereby causing more current to flow. One rule of thumb is if rise and fall times are shorter than 25 nsec, minimal current will flow. But don't go overboard. Be aware that slow transitions are more tolerable in slower operating sections because the transitions occur less often. Therefore, weigh the importance of the extra dissipation against the cost of speeding signals up.

Now Let's Reduce Power (Continued)

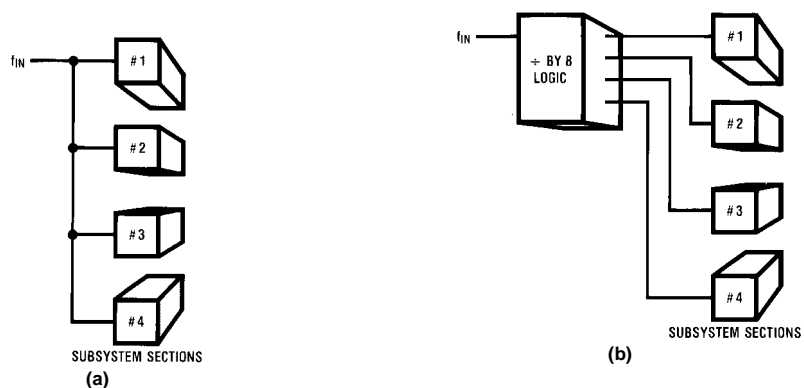


FIGURE 7. Reducing clock rate—but not throughput—this scheme allows you to reduce power by clocking a system's n subsections only as fast as needed, instead of clocking all system blocks at the full clock frequency

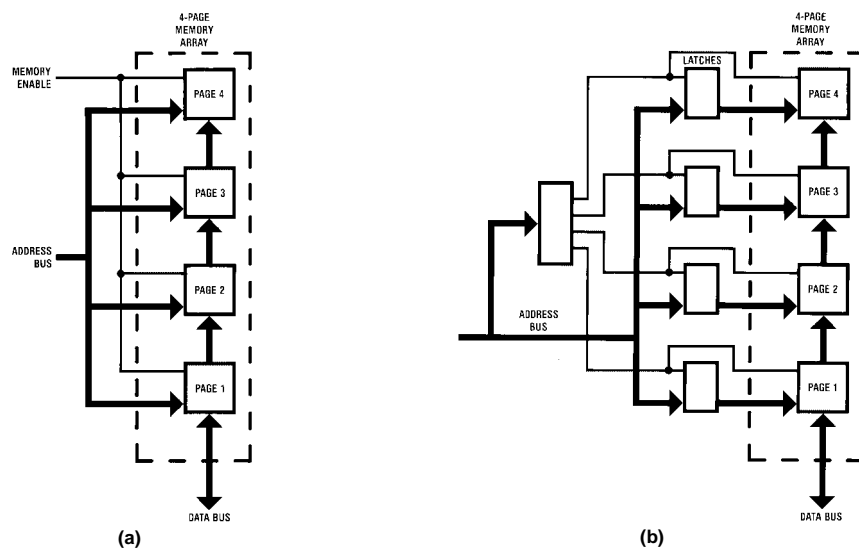


FIGURE 8. Latching memories' addresses (b) can reduce system power. In (a), every memory is driven at the bus frequency. By contrast, in (b)'s configuration, only the memory block being addressed is clocked.

Now Let's Reduce Power (Continued)

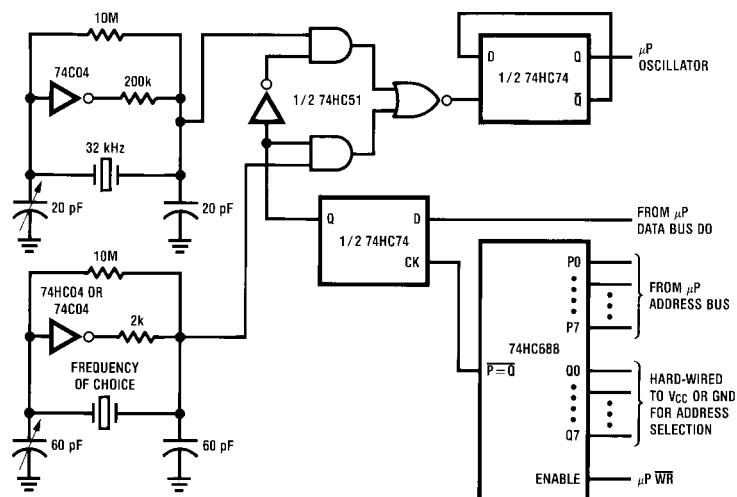


FIGURE 9. Switch your system's clock frequency for reduced power consumption. The circuit shown is a software-selectable oscillator for a microprocessor system.

It's important to point out that floating inputs can result in unnecessary power dissipation. If inputs are open, the input voltage can float to an indeterminate and intermediate level; thus, don't float CMOS Inputs. This action can turn on both p-channel and n-channel transistors, resulting in supply-current drain. In bus-oriented systems, don't allow the bus to become completely 3-STATE or float for extended periods because this will have the same effect as leaving inputs open.

Bus structures subject to prolonged 3-STATE conditions should be terminated to ensure that the bus lines pull to

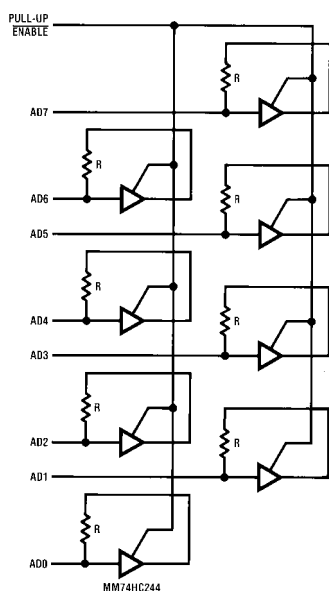
either V_{CC} or ground. For short durations, the bus capacitance can maintain valid logic levels, so for short-time floating, pull-up or pull-down resistors might not be necessary.

Finally, make sure your design ensures solid V_{CC} and ground logic levels at MM74HC inputs. If the logic LOW is greater than 0.5V or the logic HIGH is lower than $V_{CC}-0.5V$, then the normally Off p-channel or n-channel transistor can actually conduct slightly, causing additional I_{CC} to flow (similarly to the previously discussed HCT "soft" levels).

Be Wary of Static Loads

Previous sections discussed the effects of capacitive loads on system power dissipation. What about resistive loads? In ultra-low-power systems, their contribution can be significant, so it's important to find ways to eliminate or minimize their detrimental effects. The loads could be pull-up resistors, bus terminators, displays, relays or peripheral drivers. Of course, the most obvious way to reduce power is to select low-power relays or displays, for example, and to make resistor values as high as possible. In addition, you can switch these loads out of the circuit when not needed.

Figure 10 shows a circuit that dissipates no static power; you can use it to terminate a 3-STATE bus to the last active logic level seen on the bus. This technique is useful to ensure the bus doesn't float when 3-STATEd. The circuit



$100\ \Omega < R < 200k$

FIGURE 10. Dissipating zero static power, this scheme can serve to terminate a 3-STATE bus to the last active logic level seen on the bus. To disconnect the terminating resistors, use the 3-STATE Enable command.

uses a MM74HC244 whose input is tied to its output. If the terminating resistors must be completely turned off, use the 3-STATE Enable.

Figure 11 illustrates a method of controlling a series of pull-up resistors using the output of an HC gate or 3-STATE buffer. Because HC outputs can pull up to V_{CC} , you can use them as an enable for many pull-ups, as long as the parallel combination of the pull-up resistors exceeds $2\ k\Omega$. You can also use the method for pull-down resistors.

When considering whether you should add circuitry to disable pull-up resistors, remember that in CMOS systems, the pull-ups only dissipate power when the driving output is low. No power is consumed when the driving output is HIGH or at the 3-STATE level (disabled).

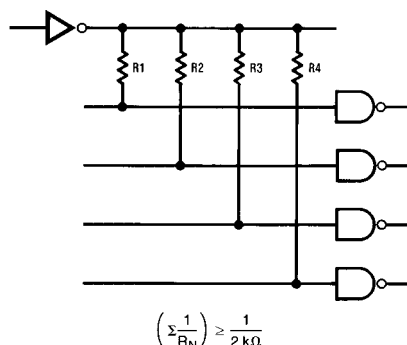


FIGURE 11. Enable or disable pull-up resistors with this configuration. You can use an HC device's outputs to enable several pull-ups. The scheme is also applicable to pull-down resistors.

The Final Solution: Power Down

When all else fails, the best way to reduce system power is to shut off the system or unnecessary parts. Before you do this, keep in mind that turning off the clock to a section of the system is almost tantamount to turning the section off (thanks to the ICs' low leakage currents). The advantage of the clock-killing approach? It avoids the complications of the power-down methods that follow.

Still, there are occasions in which parts of a system are powered down. When all or part of a system is shut off, or when one of several interconnected systems is powered

down, you should respect several criteria to avoid spurious signals during the power-down period, and to eliminate possibly fatal conditions.

One condition that requires very careful consideration is the application of high-level signals to unpowered HC devices. Figure 12(a) shows in block form the basic concepts of powering down part of a system. In this scenario, it's possible to apply a logic One to the unpowered CMOS logic. If this happens to either an input Figure 12(b) or a 3-STATE output Figure 12(c), the device will still be powered.

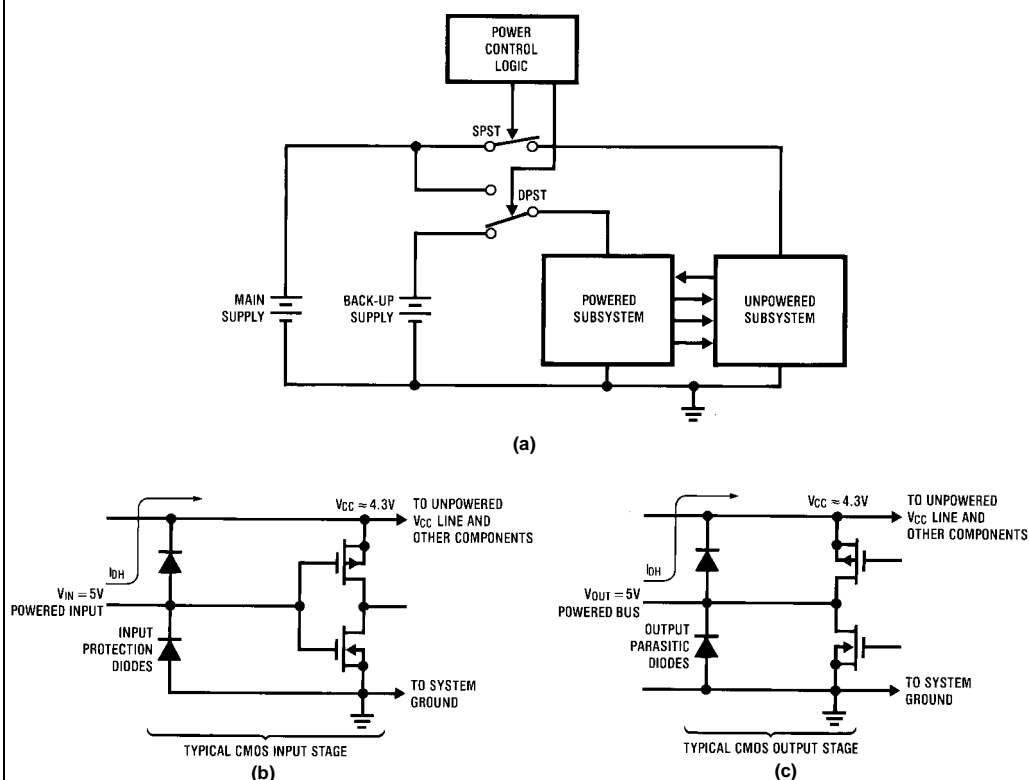


FIGURE 12. This basic power-up and -down system (a) presents dangers to CMOS-logic ICs. As the input (b) and output (c) schematics show, a logic One can actually power up the "unpowered" system, thereby causing damage to input and output diodes.

The Final Solution: Power Down (Continued)

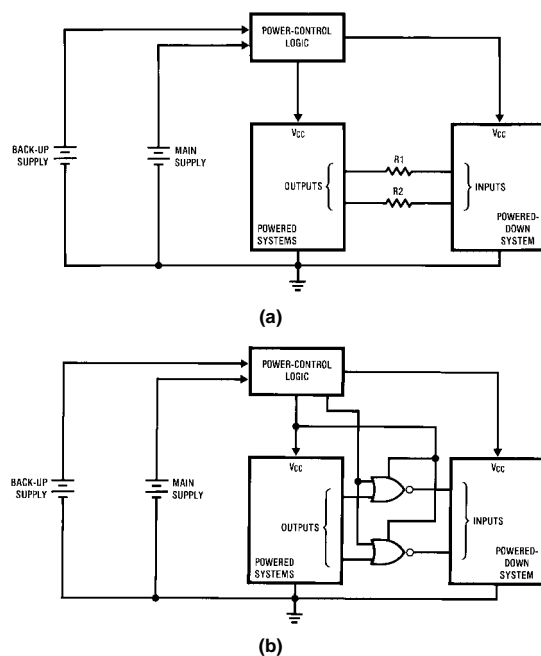


FIGURE 13. Solutions to the problems in Figure 12, these configurations protect CMOS circuits' inputs and outputs in power-down situations. The brute-force solution in (a) limits input currents; (b)'s scheme forces inputs to ground.

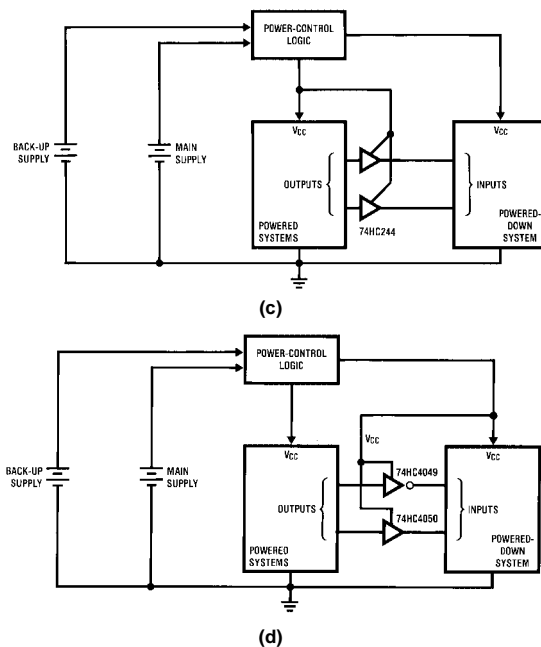


FIGURE 14. Solutions to the problems in Figure 12, these configurations protect CMOS circuits' inputs and outputs in power-down situations. The brute-force solution in (c), 3-STATE gates disable the inputs. In (d), MM74HC4049 or -4050 level translators isolate inputs from the power supply.

The Final Solution: Power Down (Continued)

Referring again to Figure 12(b) and Figure 12(c), the input protection diodes and the output parasitic diodes form a path to the V_{CC} pin. The voltage at this pin will be $V_{IN}-0.7V$. The "unpowered" system is really powered up by the logic signal through these diodes. If the "unpowered" V_{CC} line accepts appreciable current, diode damage can (and usually does) result.

Figure 13 shows several solutions to the signal-powered "unpowered" problem. A resistor in series with each input (a) limits the current to 20 mA max. This low-cost, brute-force solution has the undesirable tendency, however, to dissipate power from the supply.

To avoid extra power consumption, you can use the methods in Figure 13(b) to Figure 13(d). Upon removal of power, additional logic can force all inputs to ground (b). Alternatively, 3-STATE logic can disable the signals by presenting an open circuit (c). The third possible solution (d) is to use a MM74HC4049 or MM74HC4050—circuits that lack a V_{CC} diode. In this case, even when power is removed the inputs are isolated from the power supply.

A situation analogous to the previous section's might occur on bidirectional buses or in "party-line" media, where 3-STATE output devices are powered down on the bus. In this case, power down all but the 3-STATE buffer, as shown in Figure 15. Because the buffers inputs are shut off, the IC draws negligible extra power.

In addition to ensuring that power-down proceeds smoothly, it's important to guarantee that spurious signals from the subsystem that's shutting down do not cause logic errors in the powered section. For example, battery-backed memory must be controlled to prevent spurious writes by the host processor that's shutting off.

Figure 16 illustrates a method for eliminating spurious operation upon loss of power. First, the system detects the loss of system power prior to the system's malfunction by comparing the system voltage to an arbitrary minimum voltage (V_2), or by directly monitoring the ac line for loss of 50 or 60 Hz. Having detected this loss, the system should perform all bookkeeping operations to prepare for power-down before the minimum correct operating voltage (V_3) is attained.

At V_3 , the system cannot be guaranteed to function correctly—therefore, powered logic should disable all signals that might affect the powered or battery-backed subsystem. Once stable power is restored to the minimum operating voltage (V_4), the signals should be re-enabled.

Clearly, there is more to shutting off a system (while leaving part of it powered by a backup battery) than just switching the power supplies. The primary design consideration when powering down a system is to ensure that spurious signals do not destroy valuable data or logic conditions in the battery-operated subsystem.

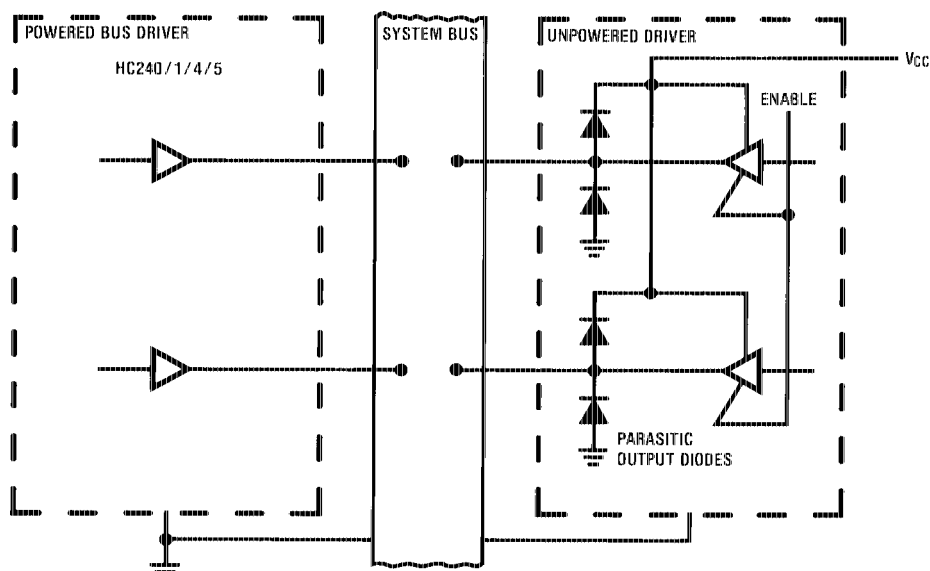
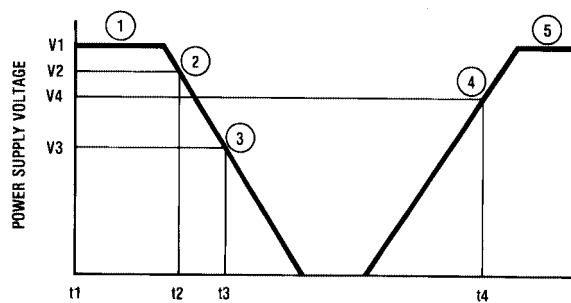


FIGURE 15. Powering down all but the 3-STATE buffer, this method protects CMOS ICs' output-protection diodes. The buffers draw negligible system power.

The Final Solution: Power Down (Continued)



TIME →

- ① NORMAL SYSTEM OPERATION
- ② LOW-VOLTAGE POWER-FAIL DETECT
- ③ MINIMUM OPERATING VOLTAGE
- ④ MINIMUM RESTORED OPERATING VOLTAGE
- ⑤ NORMAL SYSTEM OPERATION

FIGURE 16. Prevent spurious host-processor write operations in battery-backup systems by using this graph's concepts. To sum up, the system should prepare itself for power-down before minimum-operating voltage V_3 is reached, and re-enable signals when V_4 is attained upon power-up.

Fairchild does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied and Fairchild reserves the right at any time without notice to change said circuitry and specifications.

LIFE SUPPORT POLICY

FAIRCHILD'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF FAIRCHILD SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and (c) whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component in any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

www.fairchildsemi.com